Online GMM Clustering and Mini-Batch Gradient Descent Based Optimization for Industrial IoT 4.0

Seifeddine Messaoud, Abbas Bradai, *Member, IEEE*, and Emmanuel Moulay

***Abstract*—The future fifth-generation (5G) networks are expected to support a huge number of connected devices with various and multitude services having different quality of service (QoS) requirements. Communication in Industry 4.0 is one of the flagships and special applications of the 5G due to the specificity of the industrial environment as well as the variety of its services such as safety communication, robot's communications, and machine monitoring. In this context, we propose a new resource allocation for the future Industry 4.0 based on software-defined networking and network function virtualization technologies, machine learning tools and the slicing paradigm where each slice of the network is dedicated to a category of services having similar QoS requirement level. In this article, the proposed solution ensures the allocation of the resources to the slices depending on their requirements in terms of bandwidth, delay, and reliability. Toward this goal, our solution is performed in three main steps: first, Internet of Things (IoT) devices assignment to the slices step based on online Gaussian mixture model clustering algorithm, second, inter-slices resources reservations step based on mini-batch gradient descent, and third, intra-slices resources allocations based on the max-utility algorithm. We have performed extensive simulations in a realistic industrial scenario using NS3 simulator. Numerical results show the effectiveness of our proposed solution in terms of reducing packet error rate, energy consumption, and in terms of increasing the percentage of served devices in delay comparing to the traditional approaches.**

***Index Terms*—Industry 4.0, interslice, intraslice, machine learning, network slicing, network function virtualization (NFV), quality of service (QoS), software-defined networking (SDN).**

I. Introduction

With the development of wireless communication technology, the fifth-generation (5G) wireless networks are expected to support more than 50 billion connected devices and machines by 2020 [1]. Therefore, the increased number of generated data and connectivity of smart devices, which are the main driver of the Internet of Things (IoT) paradigm, are rapidly expanding its benefits in industrial environments. This provides the opportunity to establish industrial cyber-physical system via modern and reliable information exchange technologies and data digitalization [2].

This evolution is known by the fourth industrial revolution thanks to the emergence of information and communication technologies, such as cloud computing, IoT, and network softwarization and virtualization, which can be integrated with a multitude of manufacturing processes [3].

Following the industrial revolution and the explosive spread of the IoT paradigm, the industrial environment is continuing to support an increasing number of connected objects, such as robot, actuators, machines, and sensors. However, this continuous evolution will bring a radical change in the industrial landscape, allowing new production models and promising commercial opportunities. Furthermore, today's industrial networks have been designed for static manufacturing processes where changes and configurations in the manufacturing workflow require lengthy maintenance operations. In addition, different traffic flows with a multitude and various (QoS) parameters are transmitted over heterogeneous industrial environments sharing the same physical resources. This will increase cost and data loss, and will not meet the user's QoS requirements. New methodologies and automation mechanisms in the next industrial generation are required to remotely verify the management process, check the QoS, and support new challenges such as improved user's QoS, resource efficiency, and reducing cost.

On the other hand, to support flexibility and sustainability expected in next-generation manufacturing processes, industrial networks would be faced with a process of transformation. In this context, the 5G wireless network system is considered as a key enabler in this new trend by extending the network slicing paradigm to meet the drastic user's requirements, over heterogeneous industrial domains, which represents a revolutionary paradigm for industrial challenges in the coming years.

In this fervent area, network slicing is one of the most powerful solutions in industry 4.0 that can bring network drastic improvements and fulfill diverse network requirements based on the unified physical infrastructure and sharing network resources between slices. Supported by network slicing, physical resources can be dynamically reserved and allocated to logical network slices according to the corresponding QoS demands.

Manuscript received August 24, 2019; accepted September 12, 2019. Date of publication October 4, 2019; date of current version January 14, 2020. Paper no. TII-19-3882. *(Corresponding author: Abbas Bradai.)*

S. Messaoud is with the University of Monastir Laboratory of Electronics and Microelectronics, Sciences Faculty of Monastir, Monastir 5019, Tunisia (e-mail: seifeddine.messaoud@fsm.rnu.tn).

A. Bradai and E. Moulay are with the University of Poitiers XLIM-CNRS, Bât SP2MI, 86962 Chasseneuil Cedex, France (e-mail: abbas.bradai@univ-poitiers.fr; emmanuel.moulay@univ-poitiers.fr).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2019.2945012

Under the network slicing mask, software-defined networking (SDN), and network function virtualization (NFV) have been widely accepted as promising technologies. These mechanisms can improve decision-making and scalability by efficiently sharing network resources to network slices with different QoS types, managing multiple virtual networks, and dynamically configuring parameters [3].

In this context, SDN is defined as a network concept that enables centralized and intelligent control and management that can on-demand modify traffic flows according to industrial application requirements by decoupling data and control traffic. While NFV is the technology that allows the virtual representation and deployment of logical network functions as virtual network functions (VNFs). When these technologies associated with cloud computing, it can provide better solutions to support reliability and latency requirements, which still represents a major challenge of industrial 4.0 applications.

In that regard, great efforts have been carried out to integrate efficiently SDN, NFV, and network slicing based-architectures within industrial domains or within another field. Li *et al.* [4] have proposed an industrial network based on SDN mechanism in order to support dynamic production processes. Unlike traditional industrial networks, remarkable energy saving is achieved. A resource allocation method with consideration of interference management was proposed in [5], where different QoS requirements were guaranteed by optimizing jointly power and sub-channel allocation. Yazici *et al.* [6] have discussed a 5G network architecture scheme based on SDN to allocate physical resources to virtual slices within a local area and to perform scheduling among slices. An end-to-end network slicing methodology was proposed by Li *et al.* [7] in order to share horizontally physical resources whose main purpose is to create multiple virtual networks that can support industry applications. For interested readers, several architectures based on SDN technology for industrial networks and cyber-physical systems have been proposed recently in [8]–[10] and references therein. However, the majority of these proposed works do not deal with real and heterogeneous scenarios with various QoS requirements where different IoT devices or industrial machinery are operating in the same network.

Therefore, unlike previous works, we seek in our article to extend network slicing benefits in industry 4.0 by considering QoS classes using machine learning tools. The main contributions of this article are as follows.

1) We propose SDN and NFV based-network slicing architecture for industry 4.0 to meet various services requirements with guaranteed QoS for connected devices.
2) We propose an adaptive and online machine learning algorithm to learn devices requirements and assign each device to the slice that meets its QoS requirements.
3) We propose a machine learning based-dynamic inter-slicing algorithm; to split radio resources and reserve channels to slices based on required throughput.
4) We propose an intra-slicing algorithm in order to allocate dynamically resources on gateways for assigned devices; load and reliability are considered in each gateway.

The rest of this article is organized as follows. Section II is allocated to the proposed slicing architecture and optimization problem formulation. We introduce in Section III the proposed slicing approach. In Section IV, we provide a discussion about simulation results and comparisons. Finally, Section V concludes this article.

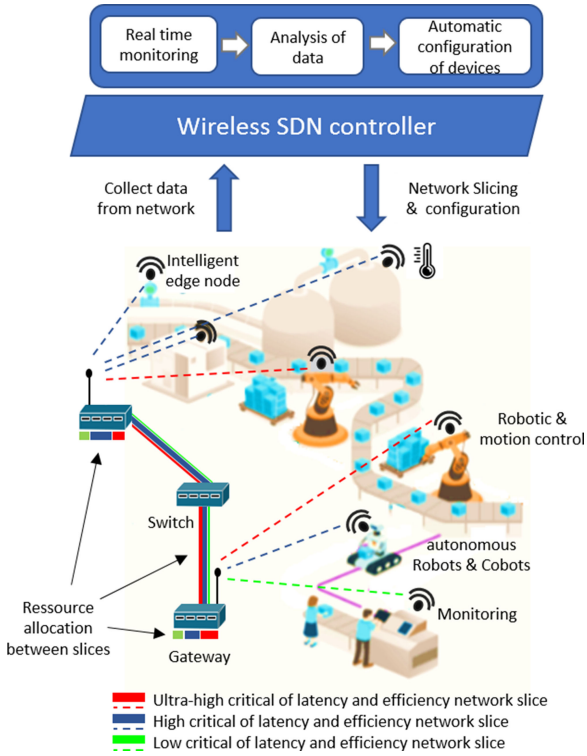

Fig. 1. Network slicing architecture for industry 4.0.

II. Proposed Network Slicing Architecture and Problem Formulation

A. Proposed Network Slicing Architecture

The 5G network architecture design should be built on a deep consideration of hardware infrastructure, software control, and the interconnectivity between them. The network slicing, which can satisfy multiple service requirements based on the unified physical infrastructure and sharing the same physical resources, is considered as a critical paradigm by providing multiple instances that operate independently for a specific network functions.

The network slicing-based 5G system architecture is given in Fig. 1. The aim of this architecture is to support the creation, control, and management of multiple network slices over factory infrastructures in order to provide high levels of flexibility and scalability and meet QoS requirements for robots, sensors, and actuators in industrial networks.

The infrastructure layer contains all the physical resources needed to perform virtualized industrial processes. We emphasize that the involved resources go beyond traditional data centers. It includes industrial equipment with sensing and actuation capabilities in addition to the physical computing, storage, and network components. The virtualization layer includes the tools and technologies required to provide a virtualization environment for hosting VNF instances. While the slicing layer refers to

the deployed industrial slices in order to accommodate specific industrial machinery's QoS. At this level, we define a set of $L = \{l_j, \ldots, l_L\}$ slices based on throughput R (with bandwidth λ_r), transmission delay D, and urgency factor ∂. Each slice is responsible for serving a set of $N = \{n_i, \ldots, n_N\}$ assigned IoT devices through a set of $G = \{g_k, \ldots, g_G\}$ gateways. Where $\theta_{i,j,k}$ denote the binary value that represent the assignment success of devices i to the slice j through gateway k. All these layers interact with the general administrative control unit represented in the SDN controller, which can control industrial networks in a centralized fashion. By interacting with the NFV management and orchestration, SDN controller is in charge of guaranteeing several industrial QoS constraints. It can acquire and allocate virtual resources for slices and enable industrial network reconfigurability, by exploiting the generated virtual flows from devices, in order to meet dynamically industrial devices QoS changes. Assume that the assigned device i to the slice j generate a virtual flow $f_{i,j,k}$ that goes from the gateway k to the SDN controller and is characterized by a utility metrics $U_{i,j,k}$. SDN controller must define also the slice's request, compute the available resources, and serve slices requirements, in a way that avoids resources starvation. At this level, we denote by $C_{j,k}$ the requested physical resources for the slice j on gateway k, while ς_k is the total gateway capacity.

In this article, we aim to jointly optimize industrial machinery QoS and network energy efficiency by providing dynamically slice members with the requested physical resources.

B. Multi-Objective Optimization Model Formulation

The purpose of this article is to optimize the performance of the industrial network in terms of QoS and energy consumption (EC). Toward this goal, the network slicing optimization concept consists of two main steps. First step is to find the best inter-slicing resource reservation strategy. This will impose challenges in aggregating necessary parameters, reconfiguring slices, and updating reserved resources. Second step is to establish the best intra-slicing resource allocation strategy. This will bring other challenges in reconfiguring devices and updating resource allocation. These problems are formulated into multi-objective functions as follows.

1) QoS Model: We define the QoS type in each slice based on the data rate R and the transmission delay D. The data rate model defined in (1) is denoted by the ratio between the peak rate χ and the slice members n_j. The peak rate of each IoT device depends on many factors such as the received power P_r, the transmit power P_t, and the received SINR, which are written in (2)–(4), respectively, [11], [12]

$$R_{i,j,k} = \frac{\chi}{n_j} \tag{1}$$

$$P_{r_{i,j}} = \frac{P_t}{L_0} d^{-\delta} h \tag{2}$$

$$P_{t_{i,j}} = \frac{\mathrm{SNIR}\,(P_I + P_N)}{L} \tag{3}$$

$$\mathrm{SINR}_{i,j} = \frac{P_r}{P_I + P_N} \tag{4}$$

where $R_{i,j,k}$ represent the data rate for an industrial IoT device i assigned to the slice j on gateway k, P_I and L are, respectively, the receiver noise and the path loss, L_0 is a constant, which depends on the antenna gains and transmission frequency, h denotes the random variable that represents the channel fading, and finally, d and δ denotes, respectively, the distance between the trans-receiver and the path loss exponent.

Moreover, the transmission delay $D_{i,j,k}$ of a device assigned to the slice j is expressed as in (5), where $M_{i,j,k}$ is the packet length that is trans-received from a device i

$$D_{i,j,k} = \frac{M_{i,j,k}}{R_{i,j,k}}. \tag{5}$$

Based on what was previously mentioned, the QoS cost is modeled by the following equation:

$$\mathrm{QoS}_{i,j,k} = \overline{R_{i,j,k}} + \left(1 - \overline{D_{i,j,k}}\right)$$
$$\max \sum_{i \in N} \mathrm{QoS}_{i,j,k} \quad \forall j \in L, k \in G \tag{6}$$

where $\mathrm{QoS}_{i,j,k}$ is the reimbursements that should be maximized at each slice and at each gateway. In addition, $\overline{R_{i,j,k}}$ and $\overline{D_{i,j,k}}$ are adopted as normalized values that denote the throughput and the transmission delay (TD) achieved by industrial connected device, respectively.

2) EC Model: The EC model is considered as the required power to trans-received a data packet, which depends on the received power and the transmitted power given in (2) and (7), respectively. In addition, the connected device consumes P_0 power by the communication module. In this context, the active and the sleep mode, as in (8), are the two energy states of an IoT device that need to be considered [11], [12]

$$P_{t_{i,j}} = \frac{P_{r_{i,j}} L_0}{d^{-\delta} h} \tag{7}$$

$$E_{i,j,k} = \left[\eta P_{t_{i,j}} + P_0\right] T_{\mathrm{active}} + \left[\eta P_{t_{i,j}} + P_0\right] T_{\mathrm{sleep}}$$
$$\min \sum_{i \in N} E_{i,j,k}, \forall j \in L \; \forall k \in G \tag{8}$$

where η denotes the electric-to-RF power conversion factor. $E_{i,j,k}$ represents the ECs for the device i assigned to the slice j that should be minimized for each slice.

In addition to the energy and QoS factors, the packet error rate (PER), formulated in (9), is an interesting factor which reflects the efficiency and reliability of the devices in each slice

$$\mathrm{PER}_{i,j,k} = \left(\frac{S_p}{T_p}\right) \times 100$$
$$\min \sum_{i \in N} \mathrm{PER}_{i,j,k} \; \forall j \in L \; \forall k \in G \tag{9}$$

where S_p denotes the number of successful packets and T_p is the number of the total packet sent. PER is the other cost function that should be minimized in order to guaranteed efficiency and reliability in each slice.

3) Network Slicing Problem Formulation: In this article, network slicing optimization problem consists of three steps. The first one involves the admission and the assignment of devices to the desired slice. The second step is the dynamic interslicing

resources reservation. While the third step is the intra-slice resource allocations. First, we define slices based on urgency factor, delay, and throughput, and then we look to assign devices to the slice that meets its QoS requirements. It is remarkable that the urgency factor, delay and throughput are the key clues for defining the device priority. Furthermore, there are other factors must be taken into consideration such as PER, reliability and the huge load rate of devices to the network. Second, we search to estimate the needed inter-slice capacity $C_{j,k}$ based on the required throughput. The purpose of the last step is to optimize the intra-slice resource allocation for each slice member. Therefore, the multi-objective optimization for the slicing and the resources allocation problem is formulated in the following equation:

$$\min \sum_{i \in N} \theta_{i,j,k} E_{i,j,k} / \mathrm{QoS}_{i,j,k} \quad \forall j \in L \, \forall k \in G \tag{10}$$

subject to the following constraints:

$$\sum_{i \in N} \theta_{i,j,k} R_{i,j,k} \langle R^{\max}_{i,j,k} \quad \forall j \in L, k \in G \tag{11}$$

$$\lambda_{r_{j,k}} \cap \lambda_{r_{j',k}} = \emptyset \quad \forall j \in L, k \in G \tag{12}$$

$$0 \langle P_{t_{i,j,k}} \langle P^{\max}_{t_{i,j,k}} \, \forall j \in L \quad \forall k \in G \, \forall i \in N \tag{13}$$

$$\theta_{i,j,k} \in \{0,1\} \, \forall j \in L \quad \forall k \in G \, \forall i \in N \tag{14}$$

where constraint (11) guarantees that the sum of transferred traffic by devices i assigned to the slice j should not exceed the maximum allocated data rate capacity. Moreover, without violating the non-interference principle between slices, constraint (12) ensures a perfect isolation between them. In other words, each slice has its own bandwidth, whether it was reserved on the same gateway or on different gateways. In addition, each device consumes a transmission power to transfer its traffic data, which should not exceed the maximum transmission power; this is provided by constraint (13). Furthermore, the binary assignment value of a device i to the slice j on the gateway k is assured by constraint (14).

III. Proposed Slicing Approach

The proposed industrial network slicing-based resource allocation scheme consists of three main steps. First, by using the online Gaussian mixture model clustering (OGMMC), each device is assigned to the slice that meets its QoS requirements. At the end of this step, the mean throughput for each slice will be estimated. Second, radio resources will be dynamically reserved for each slice based on the dynamic mini-batch gradient descent (MBGD) algorithm. Finally, the preserved radio channels for each slice will be dynamically allocated to the slice members based on the max-utility intraslice resource allocation algorithm.

A. IoT Devices Assignment and Mean Throughput Estimation: Online GMM Clustering Algorithm

Due to the ultra-diversity of industrial 4.0 services, slices are defined based on urgency, energy, and efficiency requirements to meet their objectives. Our proposed architecture is composed of three virtual industrial slices. The first slice called "Ultra-high Critical of Latency and Efficiency (UCLE)" which has the most interest slicing priority and gives more importance to the QoS, efficiency, and reliability. This makes it required by several industrial IoT applications for safety such as emergency action and safeguarding systems. Where "High Critical of Latency and Efficiency (HCLE)" slice gives a less prominence to the latency and considers reliability as a first target. This service is required by the scale readings applications. The last slice is the "Low Critical of Latency and Efficiency (LCLE)," which has the lowest slice priority with non-guaranteed QoS and efficiency. Table I summarizes the key QoS requirements [13]–[15].

TABLE I
QoS Requirements for Industrial Slices

Slice type	Latency	Reliability	Packet size	Priority (urgency)	Applications
UCLE	50ms	$1-10^{(-6)}$	24B	1	Emergency action, safeguarding systems
HCLE	100ms	$1-10^{(-6)}$	512B	2	Scale readings
LCLE	500ms	$\rangle\rangle\, 1-10^{(-6)}$	250B	3	Standard mobile robot

After specifying each slice requirements, IoT devices will be assigned to the corresponding slices. In this context, GMM is adopted as a dynamic and online clustering method (OGMMC) to assign devices to the desired slice, by checking its QoS demands and estimating the mean throughput for slices [16].

Considering a set of J mixture multivariate Gaussian Distributions, indexed by the set of parameters $\Theta = \{\alpha_j, \theta_j\}$, where $\theta_j = \{\mu_j, \sum_j\}$ denotes the parameters of the jth Gaussian distribution, in which the mean is denoted by μ_j, $\sum_j$ is the covariance, and $\alpha_j \in \{0,1\}$ is the mixing probabilities. Assume that all devices data point $\{n_i, ..., n_N\}$ are independently and identically distributed according to the mixture probability density function $P(n_i|\theta_j)$. Θ is the parameter that will be estimated to clusters using the maximum likelihood estimation process, as in (15). While the log-likelihood is formulated in (16)

$$\begin{aligned} L(\Theta|N) &= \prod_{i \subset N} P(n_i|\theta_j) \\ &= \prod_{i \subset N} \left(\sum_{j \subset J} \alpha_j P_j \left(n_i | \mu_j, \sum_j \right) \right) \end{aligned} \tag{15}$$

$$\log(\Theta|N) = \sum_{i \subset N} \log \left(\sum_{j \subset J} \alpha_j P_j \left(n_i | \mu_j, \sum_j \right) \right). \tag{16}$$

In view of the structural complexity of (16), the optimal $\hat{\Theta}$ cannot be obtained by setting the derivatives to zero. Expectation Maximization (EM) process [17] is a powerful method used to maximize the log-likelihood function and find the optimal

parameters. The latter updates iteratively the parameters of individual Gaussian distributions. The given data are considered as incomplete data. This allows defining M latent variables $M = \{m_i, ..., m_M\}$ where each m_i indicates which Gaussian component generates the data vector n_i. The new function that should be maximized is formulated in the following equation:

$$\Psi(\Theta, \varphi) = \sum_{i \subset N} \sum_{j \subset J} \log\left(\alpha_j P_j\left(n_i \left| \theta_j \right.\right)\right) - \sum_{i \subset N} \sum_{j \subset J} \omega_{i,j} \log\left(\omega_{i,j}\right) \quad (17)$$

where $\varphi = \{\omega_{i,j}\}$ is the assignment of data vectors (devices) to the clusters. In the expectation step, EM process tends to calculate the probability of a device i belongs to the cluster j (slice), by maximizing the Ψ function over the assignment φ and by considering the posterior probability $\omega_{i,j}$ formulated in the following equation:

$$\omega_{i,j} = P\left(m_i = j \left| n_i \right., \Theta\right) = \frac{\alpha_j P_j\left(n_i \left| \theta_j \right.\right)}{\sum_{j \subset J} \alpha_j P_j\left(n_i \left| \theta_j \right.\right)}. \quad (18)$$

After that, a maximization of Ψ function over the parameter Θ will be in the maximization step. As a maximization result, the parameters Θ will be estimated and updated iteratively until the convergence. The updated mean, mixing probabilities and covariance are presented, respectively, in the following equations:

$$\hat{\mu}_j^{\text{new}} = \frac{\sum_{i \subset N} \omega_{i,j} n_i}{\sum_{i \subset N} \omega_{i,j}} \quad (19)$$

$$\hat{\alpha}_j^{\text{new}} = \frac{1}{n} \sum_{i \subset N} \omega_{i,j} \quad (20)$$

$$\widehat{\sum_j^{\text{new}}} = \frac{\sum_{i \subset N} \omega_{i,j} \left(n_i - \hat{\alpha}_j^{\text{new}}\right)\left(n_i - \hat{\alpha}_j^{\text{new}}\right)^T}{\sum_{i \subset N} \omega_{i,j}}. \quad (21)$$

The optimum slicing strategy for a limited physical capacity is to virtually reserve resources for each slice based on the mean throughput R_j^T of its members. After the assignment step, OGMMC calculates R_j^T for slices, as in (22), with respect to the urgency factor

$$R_j^T = \frac{\sum_{i \subset N} R_{i,j,k}}{n_{i,j}}, \quad \forall j \in L, \forall k \in G. \quad (22)$$

Each new device sends a connection request to the server that will set up the flag *H*, in which the Online GMMC algorithm, as in Pseudo-Code 1, will be re-executed.

B. Dynamic Inter-Slicing Resources Reservation: Dynamic MBGD Algorithm

After assigning IoT devices to the slice that meets its QoS requirements and estimating the mean throughput for slices, we seek in this section to reserve dynamically interslices channel resources. To reach this goal, MBGD [18] learning algorithm is adopted as a powerful scheme to improve QoS and minimize cost, in (10), by finding the optimal throughput parameter needed to split channel resources between slices.

Pseudo-Code 1: Adaptive OGMMC Algorithm.

Input: Set of IoT devices N, Set of Clusters J, parameter Θ, $H = 1$, convergence criterion ε.
Output: $\omega_{i,j}$, parameters Θ.

```
1:  BEGIN
2:  while H = 1 do
3:      Define clusters with initialized parameters Θ
4:      while convergence=false do
5:          for each cluster j do
6:              for each IoT device do
7:                  Assign devices to clusters (slices): (18)
                    Maximize the log-likelihood function: (17)
                    Update the parameters: (19) (20) (21)
8:                  if (Ψ − Ψ_{t+1} ≤ ε) then
9:                      Convergence ← true
10:                 else
11:                     Convergence ← false
12:                 end
13:             end
14:         end
15:     end while
16: end while
17: for each slice do
18:     Compute the mean throughput: (22)
19: end
20: END
```

In the context of industrial IoT (IIoT), we assume that slices should be ready to support and serve IoT devices. The global idea is to reserve a minimum capacity level $C_{j,k}$ for slices. Then, by checking slices requirements and learning dynamically devices throughput, more radio resources will be reserved for slices.

The adopted MBGD scheme consists of two phases; pre-learning phase and learning phase. In the pre-learning phase (line 1–7 in the Pseudo-Code 2), MBGD splits physical resources between slices, by reserving minimum radio channels based on the estimated mean throughout, even there are no assigned devices (line 4–6). With respect to the slice urgency factor, it starts by computing the slice rate γ_j based on R_j^T, as in (23). This is for defining an appropriate and optimal resource distribution strategy and for not exceeding the maximum gateway capacity. Then, the defined level of physical resources will be reserved, as in (24). At the end of this phase, the process computes the unserved capacity ξ_k on each gateway, as formulated in (25)

$$\gamma_j = \frac{R_j^T}{\sum_{j \subset L} R_j^T} \quad (23)$$

$$C_{j,k} = \gamma_j * \varsigma_k \; \forall j \in L \quad \forall k \in G \quad (24)$$

$$\xi_k = \varsigma_k - \sum_{j \subset L} C_{j,k} \; \forall j \in L \quad \forall k \in G. \quad (25)$$

Pseudo-Code 2: Dynamic MBGD Algorithm.

Input: Set of R_j^T, gateway capacity, Mini-Batch datasets, stop criterion ε,
Output: $\xi_{j,k}^{\text{new}}$, $C_{j,k}^{\text{new}}$, $R_{i,j,k}^{\text{new}}$

1: **BEGIN**
2: Sort slices in decreasing order based on ∂ (urgency factor)
3: **for** each slice l_j and GW_k **do**
4: **if** $(R_{l_j}^T = 0)$ **then**
5: Define a minimum $R_j^T = R_j^{T\min} // R_j^{T\min}$ slightly greater than zero
6: **end**
7: Define slice rate: (23)
Define and reserve capacity: (24)
Compute unserved capacity: (25)
8: **for** *each Mini-Batch* **do**
9: **while** convergence=false **do**
10: Compute the gradient**:** (30)
Update the throughput: (31)
11: **if** $\overline{R_{i,j,k}^{\text{new}}} \langle \sum_{j\subset L} \overline{R_{i,j,k}} \& \xi_{j,k} \rangle 0 \parallel (|\nabla_R - \nabla_R^{t+1}|) \leq \varepsilon$
then
12: Compute new slice rate: (32)
Compute new requested capacity: (33)
Update the total reserved capacity: (34)
Compute the new unserved capacity: (35)
Convergence $\leftarrow$ false
i = i + 1; //next iteration
13: **else**
14: Convergence $\leftarrow$ true
j = j + 1; //next slice
15: **end**
16: **end while**
17: **end**
18: **end**
19: **END**

Noting that the slice capacity $C_{j,k}$ do not exceed the maximum capacity ς_k provided by each gateway. That is to say, that the mean throughput R_j^T should not exceed the sum of requested throughput for all IoT devices. Therefore, (22) and (23) must satisfy constraints (27) and (26), respectively

$$0 \langle \gamma_j \langle 1 \tag{26}$$

$$0 \langle R_j^T \langle \sum_{j\subset L} R_j^T. \tag{27}$$

After defining a minimum capacity level for each slice, MBGD acts as a brain in each gateway. It learns QoS and EC properties in order to find the best configuration parameter that meets slice demands. This can be done by tracking dynamically slices member's throughput requirements and updates its radio resources capacity.

The commonly chosen parameter in the learning phase (line 8–17) is $R_{i,j,k}$ that can interact with QoS, EC, and PER. However, when the throughput increase, QoS will be maximized. This refers to the capacity that will be increased according to $R_{i,j,k}$ that decrease the TD as denoted in (5). On the other hand, the activation time T_{active} of the IoT device will decrease because of the higher speed transmission packets in a large bandwidth. The latter has an effect on EC by decreasing the transmission power and give more chances to increase the percentage of successful transmitted packet. This parameter will be tuned and configured online, using the mean square error (MSE) process, based on the other discussed parameters (training set). In this context, let denote by $\mathbb{C}$ the current observation values and $\mathbb{C}'$ is the trained output value formulated in (28), where ν is the learning rate. The MSE function is defined in (29), where δ_j is the mini-batch size and $b_{i,j,k} = (1 - \overline{D_{i,j,k}})$

$$\mathbb{C}' = E_{i,j,k}/\text{QoS}_{i,j,k} \tag{28}$$

$$\text{MSE} = \frac{1}{2\delta_j} \sum_{i\to\delta_j} (\mathbb{C}' - \mathbb{C})^2 = \frac{1}{2\delta_j} \sum_{i\to\delta_j} \left(\frac{E_{i,j,k}}{\overline{R_{i,j,k}} + b_{i,j,k}} - \mathbb{C} \right)^2. \tag{29}$$

MBGD repeatedly iterates through the training set (power, throughput, and delay) and update the parameter $R_{i,j,k}$ according to the gradient error, respectively, in the following equations:

$$\nabla_R = \frac{-1}{\delta_j} \sum_{i\to\delta_j} \frac{E_{i,j,k}}{\left(\overline{R_{i,j,k}} + b_{i,j,k}\right)^2} * \left(\frac{E_{i,j,k}}{\left(\overline{R_{i,j,k}} + b_{i,j,k}\right)} - \mathbb{C} \right) \tag{30}$$

$$\overline{R_{i,j,k}^{\text{new}}} = \overline{R_{i,j,k}} - \upsilon \nabla_R. \tag{31}$$

Thereafter, MBGD checks resource demands for each slice and updates slice rate and requested capacity formulated, respectively, in (32) and (33). Then, it updates reserved and unserved capacity presented, respectively, by formulas (34) and (35). As summarized in Pseudo-Code 2, the learning phase will be repeated (line 8–17) until it serves all slices requirements and stops when there are no more resources to serve it or convergence is reached. We note that notations with "*new*" means the predicted value by the proposed learning process. $C_{j,k}^{\text{add}}$ denote the new requested capacity to be added, $C_{j,k}^{\text{new}}$ is the total reserved capacity, $\xi_{j,k}^{\text{new}}$ is the new unserved capacity, while $\overline{R_{i,j,k}}$ denotes the current observation parameter

$$\gamma_j^{\text{new}} = \frac{\overline{R_{i,j,k}^{\text{new}}}}{\sum_{j\subset L} \overline{R_{i,j,k}}} \tag{32}$$

$$C_{j,k}^{\text{add}} = \gamma_j^{\text{new}} * \varsigma_k \tag{33}$$

$$C_{j,k}^{\text{new}} = C_{j,k} + C_{j,k}^{\text{add}} \tag{34}$$

$$\xi_{j,k}^{\text{new}} = \varsigma_k - \sum_{j\subset L} C_{j,k}^{\text{new}}. \tag{35}$$

Pseudo-Code 3: Max-Utility Intra-Slice Resource Allocation.

Input: Set of IoT devices, set off gateways and slices
Output: Max-Utility flows allocation for IoT-device

```
BEGIN
for each gateway do
    Put slices in increasing order based on ∂ (urgency factor) Initialize flow utilities to zero
end
for each slice do
  for each IoT devices do
    Find path with the highest utility U_{i,j,k}
    Allocate IoT device i to f_{i,j,k}
  end
end
END
```

C. Dynamic Intra-Slicing Resource Allocation: Max-Utility algorithm

After interslice resource reservation, we seek in this stage to improve and optimize intra-slice resource allocation. The latter is reached by maximizing utility metric $U_{i,j,k}$ for IIoT devices, in each slice and on each gateway.

The utility metric is modeled based on reliability weight (w_r) and load weight (w_{ld}). As mentioned previously in Table I, slices are different in terms of QoS. However, utility metrics can be expressed as follows:

$$U_{\mathrm{UCLE}} = x_r\left(w_r \vartheta_r\right), x_r = \mathrm{SINR}_{i,j,k}/\mathrm{SINR}_{\max}, x_r \in \{0,1\} \tag{36}$$

$$U_{\mathrm{HCLE}} = w_r\,\vartheta_r + w_{ld}\,\vartheta_{ld} \tag{37}$$

$$U_{\mathrm{LCLE}} = w_{ld}\,\vartheta_{ld} \tag{38}$$

where the highest required reliability and urgency for UCLE slice is denoted by (36). x_r is considered as a minimum threshold guaranteed during the search for the highest reliable link. ϑ_{ld} and ϑ_r are, respectively, the load rate and the reliability rate. The algorithm summarized in Pseudo-Code 3 based on the analytical and hierarchy process, searches for the efficient and reliable link that gives the highest utility metric and allocate resources accordingly [19].

Formula (37) represents the tradeoff between reliability and load which explains the less critical latency and priority of HCLE slice and the massive number of connected IoT devices. In this case, the process goes to find the optimal link that gives a perfect reliability with minimum load. LCLE slice modeled by (38) shows the nonguaranteed latency and QoS requirements. Here, the adopted algorithm seeks to find the virtual link without considering reliability.

In general case, we consider a set on IoT device denoted as a source node assigned to slice j, uploads traffic through gateways k. The goal is to find the efficient virtual flow $f_{i,j,k}$ that maximizes device utility metric $U_{i,j,k}$, as in (39), in order to allocate efficiently resources

$$U_{i,j,k} = U'_{i,j,k} + U''_{i,j,k} \tag{39}$$

where $U''_{i,j,k}$ and $U'_{i,j,k}$ are the utilities provided by each gateway, which depends on reliability and load.

TABLE II
SIMULATION PARAMETERS

Parameter	Value
Simulation area	1 km^2
Power consumption Tx/Rx	25 mw
Battery capacity	230 mAh
Number of Nodes	1000
Number of gateways (GWs)	1
Number of channels	8 per GW
Used protocol	LoRaWAN

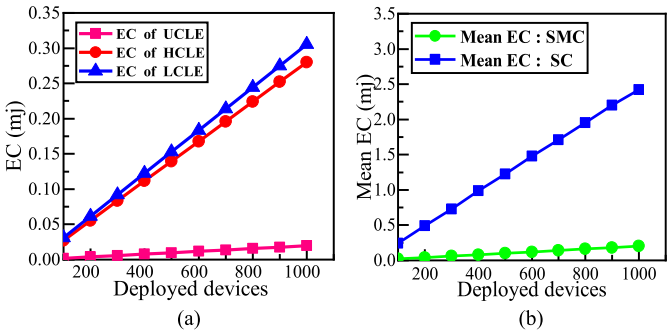

Fig. 2. EC evaluation.

IV. Simulation and Results Analysis

In this section, we study the performance of the proposed approach and deeply analyze results. The suggested scheme is implemented in NS3 simulator [20]. Table II summarizes simulation parameters.

We consider a set of IoT devices initialized with 100 devices, which increased till it reaches 1000 in a single gateway. They are distributed randomly in an industrial area of one square kilometer. We implement first the slicing methodology configuration (SMC). Then, static configuration (SC) will be implemented. The latter has the same simulation parameters to the SMC, but it does not contain the QoS constraint. The objective is to study the QoS profitability for slicing strategy results and make a comparison with the traditional configuration in terms of EC, TD, and PER.

A. EC Analysis

We assume that the sleep power for IoT devices is set to zero. As presented in Fig. 2(a), the total EC for each slice depends on the number of assigned devices and QoS configuration.

As in Table I, LCLE slice is configured with the lowest priority and nonguaranteed QoS with considering load only. As result, a large number of devices will be assigned to this slice, in which activation time will be increased proportionally. These lead to increase the power consumption. While HCLE slice scored less

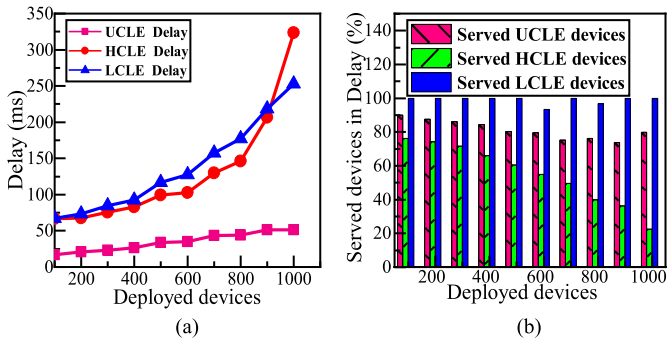

Fig. 3. Delay variation and percentage of served devices.

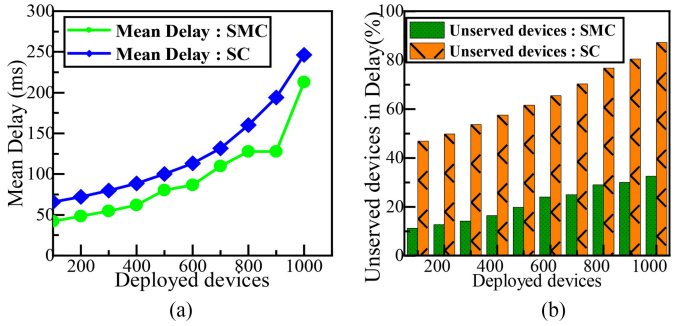

Fig. 4. Mean delay variation and percentage of unserved devices.

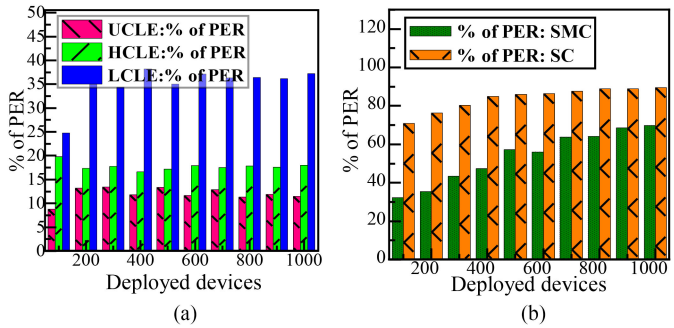

Fig. 5. Percentage of PER evaluation for both configurations.

power consumption. The latter is configured with a less critical latency but also with a guaranteed reliability and efficiency. As result, a little set of devices will respect this constraint and assigned to HCLE. The tradeoff between reliability and less critical latency leads to minimizing EC compared to the previous slice. While configuration in UCLE slice leads to the most efficient power consumption compared to the others. This returns to the inter- and intra-process that gives higher importance and priority for QoS configuration, and reliability in utility calculation. It allows the little set of assigned devices to take the most reliable gateway with smaller duration of spectrum occupation time.

For more evaluation, the mean EC for proposed scheme was compared to the EC for the SC. As denoted in Fig. 2(b), the EC for static method increases exponentially while the number of deployed devices increases. This because all IoT devices are assigned to the same network without respecting reliability, QoS, and efficiency constraint. This proves the efficiency of the proposed slicing method.

B. Delay Variation Analysis

Relying on the QoS class in Table I, each slice is configured with packet size constraint. It is remarkable that HCLE has the highest packet size than LCLE and UCLE slices. The UCLE slice is configured with the highest reliability and efficiency. This will give more chances to UCLE slice to serve its members, by maximizing utility metrics and allocating more radio channels. As result, throughput will be increased, allowing delay to be reduced to a minimum. This increases the percentage of devices that have not violated their delay threshold. This is not the case for HCLE, which is configured with medium priority, large packet size, and a utility depending on reliability and load. As the number of deployed devices increases, delay will increase exponentially, even if the learning process reserves more resources. Thus, increase the percentage of IoT devices that violated its delay threshold. Instead, delay in LCLE will be increased according to the increase of its member but it remains less than HCLE. This refers to the QoS constraint that configures LCLE with a little packet size than HCLE and considers only the load. Fig. 3(a) demonstrates the delay variation for slices, while Fig. 3(b) demonstrates the percentage of served devices in delay.

More performance evaluations were conducted, and the proposed slicing method was compared to the static method in terms of delay variation and percentage of unserved devices in delay. As seen in Fig. 4(a) and (b), SC had the worst results with highest delay varies exponentially with the increase of deployed devices number. Also 88% of devices did not respect their delay thresholds on 1000 deployed devices compared to 32% of the SMC. This refers to the random configuration that did not take into consideration QoS requirements of devices.

C. PER Analysis

Fig. 5(a) shows the evolution of PER in each slice. As described previously, the proposed process gives more importance to the UCLE slice, by checking its QoS demands and serve it. Then, it moves to the HCLE slice, and later if there are unserved resources it can reserve some to LCLE slice. As result, UCLE slice will be frequently served, and may, therefore, limit PER of its members. It is remarkable that PER in UCLE increases when devices increase at 200 and 300, then PER decrease at 400 devices, etc. This returns to the learning tools that try to avoid resource starvation in each slice and dynamically reserves channels following throughput demands. As the deployment device increases, congestion increases and reserved resources will no longer be sufficient to support devices' requirements. At this stage, PER will increase until process serves its demand in future iteration. In fact, it is the same thing for the other two slices, but with considering less QoS constraint to HCLE and no QoS constraint to the LCLE slice. This implies that fewer radio channels will be reserved, in which congestion will be

increased, while PER will also increase. On the other hand, with static configuration, PER was highly increased. This refers to the continuously increased congestion with the increase of deployment devices. This is because the QoS configuration is not considered. The results in Fig. 5(b) prove the efficiency of the optimization process in reducing PER with 40% compared to 20% of the SC.

V. Conclusion

The ever-increasing exploitation of smart devices with improved capabilities is leading to a radical change in the industrial landscape. However, low latency, reliability, and efficiency are required to support new industrial 4.0 challenges. Network slicing paradigm benefits can be extended to address outrange performance requirements. In this article, we proposed industrial network slicing framework based on machine learning tools in order to meet QoS requirements online and dynamically over the industrial network. First, OGMMC was proposed to assign devices to the desired slice that meet its QoS demands and to estimate mean throughput slices requirements. Second, mini-batch-based slicing scheme was proposed to reserve dynamically radio channels to slices. Finally, steps max-utility algorithm was adopted in order to efficiently allocate the gateway resources to the slice's members. Simulations results showed the efficiency of the proposed slicing method compared to the static method in saving EC, reducing delay, and PER.

References

[1] D. Hanes, G. Salgueiro, P. Grossetete, R. Barton, and J. Henry, *IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things*. Indianapolis, IN, USA: Cisco Press, 2017.

[2] A. Colombo, W. Karnouskos, S. Kaynak, O. Y. Shi, and S. Yin, "Industrial cyberphysical systems: A backbone of the fourth industrial revolution," *IEEE Ind. Electron. Mag.*, vol. 11, no. 1, pp. 6–16, Mar. 2017.

[3] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Commun. Surveys Tut.*, vol. 20, no. 3, pp. 2429–2453, 3Q 2018.

[4] D. Li, M. T. Zhou, P. Zeng, M. Yang, Y. Zhang, and H. Yu, "Green and reliable software-defined industrial networks," *IEEE Commun. Mag.*, vol. 54, no. 10, pp. 30–37, Oct. 2016.

[5] H. Zhang, C. Jiang, N. C. Beaulieu, X. Chu, X. Wen, and M. Tao, "Resource allocation in spectrum-sharing OFDMA femtocells with heterogeneous services," *IEEE Trans. Commun.*, vol. 62, no. 7, pp. 2366–2377, Jul. 2014.

[6] V. Yazici, U. C. Kozat, and M. O. Sunay, "A new control plane for 5G network architecture with a case study on unified handoff, mobility, and routing management," *IEEE Commun. Mag.*, vol. 52, no. 11, pp. 76–85, Nov. 2014.

[7] Q. Li, G. Wu, A. Papathanassiou, and U. Mukherjee, "An end-to-end network slicing framework for 5G wireless communication systems," 2016, arXiv:1608.00572.

[8] E. Molina and E. Jacob, "Software-defined networking in cyber-physical systems: A survey," *Comput. Elect. Eng.*, vol. 66, pp. 407–419, 2018.

[9] J. Wan, S. Tang, Z. Shu, D. Li, S. Wang, M. Imran, and A. V. Vasilakos, "Software-defined industrial Internet of Things in the context of industry 4.0," *IEEE Sensors J.*, vol. 16, no. 20, pp. 7373–7380, Oct. 2016.

[10] J. Wan, C. F. Lai, H. Song, M. Imran, and D. Jia, "Software-defined industrial Internet of Things," *Wireless Commun. Mobile Comput.*, 2019.

[11] B. Al Homssi, A. Ai-Hourani, K. G. Chavez, S. Chandrasekharan, and S. Kandeepan, "Energy-efficient IoT for 5G: A framework for adaptive power and rate control," in *Proc. 12th Int. Conf. Signal Process. Commun. Syst.*, Dec. 2018, pp. 1–6.

[12] A. Al-Hourani, S. Kandeepan, and E. Hossain, "Relay-assisted device-to-device communication: A stochastic analysis of energy saving," *IEEE Trans. Mobile Comput.*, vol. 15, no. 12, pp. 3129–3141, Dec. 2016.

[13] A. E. Kalør, R. Guillaume, J. J. Nielsen, A. Mueller, and P. Popovski, "Network slicing in industry 4.0 applications: Abstraction methods and end-to-end analysis," *IEEE Trans. Ind. Informat.*, vol. 14, no. 12, pp. 5419–5427, Dec. 2018.

[14] M. Lucas-Estañ, M. Sepulcre, T. Raptis, A. Passarella, and M. Conti, "Emerging trends in hybrid wireless communication and data management for the industry 4.0," *Electronics*, vol. 7, no. 12, 2018, Art. no. 400.

[15] A. E. Kalør, R. Guillaume, J. J. Nielsen, A. Mueller, and P. Popovski, "Network slicing for ultra-reliable low latency communication in industry 4.0 scenarios," *arXiv:1708.09132*, 2017.

[16] G. Celeux and G. Govaert, "Gaussian parsimonious clustering models," *Pattern Recognit.*, vol. 28, no. 5, pp. 781–793, 1995.

[17] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal Process. Mag.*, vol. 13, no. 6, pp. 47–60, Nov. 1996.

[18] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv:1609.04747*, 2016.

[19] S. Dawaliby, A. Bradai, and Y. Pousset, "Adaptive dynamic network slicing in LoRa networks," *Future Gener. Comput. Syst.*, vol. 98, pp. 697–707, 2019.

[20] G. Carneiro, "NS-3: Network simulator 3," in UTM Lab Meeting, vol. 20, pp. 4–5, Apr. 2010.